Slinky 

CS 402
Dr. Henderson
Fall 2025
Version 1

For this assignment you will be practicing coding asynchronous events.

You will be building a web-crawler that extracts links from a web-page and recurses into each link up to a specified depth.

Your program must print out a sorted (alphabetically) list of extracted links to the console. The list should not contain any duplicates.

The command line must take a `--depth` (aliased to `-d`) option which takes an integer representing the maximum levels to recurse (default must be 5).

The initial starting url is specified on the command line after the options.

Example:

```
bin/slinky.exe -d 1 https://google.com

http://www.google.com/history/optout?hl=en
https://accounts.google.com/ServiceLogin?
hl=en&passive=true&continue=https://www.google.com/&ec=GAZAAQ
https://drive.google.com/?tab=wo
https://google.com
https://mail.google.com/mail/?tab=wm
https://maps.google.com/maps?hl=en&tab=wl
https://news.google.com/?tab=wn
https://play.google.com/?hl=en&tab=w8
https://www.google.com/imghp?hl=en&tab=wi
https://www.google.com/intl/en/about/products?tab=wh
https://www.youtube.com/?tab=w1
```

See the comments in the source modules for the requirements.

Notes on the crawlPage function:

- use the `http` package to fetch the html document: `final response = http.get(Uri.parse(url))`
- use the `html/parser` package to parse the body of the html document: `final document = parse(response.body)`
- use the `Document` object method `querySelectorAll()` to get a list of HTML anchor (link) elements: `final anchors = document.querySelectorAll('a')`
- each anchor element will have an `href` attribute which is the link url: e.g. `anchors[0].attributes['href']`
- discard any `href` strings that don't start with `http` (these are relative links we

don’t care about)

General notes

- You will need to determine which functions are asynchronous and use `Future` objects to wrap them.
- You will need to use the `await` operator to “unwrap” any Futures.
- You can use the `Future.wait()` method to wait (and “unwrap”) a list of Future objects.
- Remember any function that uses the `await` operator must be declared as `async`